

**REMARKS**

Favorable reconsideration of this application is respectfully requested in view of the claim amendments and following remarks. Claims 1-28 are pending in the present application of which claims 1 and 13 are independent. Claim 1 has been amended. Claims 25-28 have been added. No new matter has been added.

Claims 1-12 stand rejected under 35 U.S.C. § 101 as allegedly being directed to non-statutory subject matter. Claims 1-3, 6-9, 12-15, 18-21 and 24 stand rejected under 35 U.S.C. § 102(e) as allegedly being anticipated by Loginov (U.S. Patent No. 6,567,831) (“Loginov”). Claims 4, 5, 10, 11, 16, 17, 22 and 23 stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over Loginov in view of Aho et al. (“Compilers: Principles, Techniques, and Tools”, ISBN 0-201-10088-6) (“Aho”). These rejections are respectfully traversed for at least the following reasons.

**Drawings**

At the outset, the indication that the drawings filed on September 28, 2001 have been accepted is noted with appreciation.

**Objection to the Specification**

The Examiner required correction of an informality on page 4, col. R:6 of the published application, which corresponds to page 13, line 22 of the original filed specification. The Applicants have amended the specification accordingly to correct the informality. No new matter has been added.

*Claim Objections*

The Examiner objected to claims 2-4, 6, 8, 14-16 and 18 because the recitation of “and” in these claims should be --or--. The Examiner states that the recitation of “includes at least one of trigonometric, hyperbolic, *and* square root functions” in claim 6 should be corrected to recite “includes at least one of trigonometric, hyperbolic, *or* square root functions.” The Applicants respectfully submit that the objection is improper. As set forth in MPEP 2173.05(h), “alternative expressions are permitted if they present no uncertainty or ambiguity with respect to the question of scope or clarity of the claims.” The Applicants submit that the recitation of “and” in claims 2-4, 6, 8, 14-16 and 18 presents no uncertainty or ambiguity with respect to the question of scope or clarity of the claims. However, recitation of “or” in claims 2-4, 6, 8, 14-16 and 18 would present uncertainty and ambiguity with respect to the scope and clarity of the claims. For example, recitation of “includes at least one of trigonometric, hyperbolic, *or* square root functions” would change the scope of claim 6 to include *only one of* trigonometric, hyperbolic, or square root functions, rather than “at least one of” the recited functions. As currently recited in claim 6, “includes at least one of trigonometric, hyperbolic, *and* square root functions” encompasses at least one, e.g., one, two, or three of the recited functions, which is broader in scope than recitation of “or” in the claim. The Applicants therefore respectfully request withdrawal of the objection.

*Claim Rejection under 35 U.S.C. 101*

The test for determining whether an invention is directed to statutory subject matter under 35 U.S.C. § 101 is whether the claimed invention as a whole accomplishes a practical application (MPEP 2106). As noted by the Court of Appeals for the Federal Circuit in *State Street*, 47 USPQ2d at 1601-02 (Fed. Cir. 1998), the claimed invention must produce a “useful, concrete and tangible result.”

Claims 1-12 stand rejected under 35 U.S.C. § 101 as allegedly being directed to non-statutory subject matter. This rejection is respectfully traversed.

Claim 1, as amended, now recites a compiler *used by a computer architecture* to compile a family of related functions, comprising “a member recognizer configured to recognize a member function from said family of related functions,” “a family start caller configured to make a family-start function call for said family of related functions” and “a member finish caller to make a member-finish function call for said member function.”

The Applicants submit that the claimed “compiler *used by a computer architecture* to compile a family of related functions” as recited in claims 1-12 is directed to statutory subject matter under 35 U.S.C. § 101.

The functions of the claimed compiler have practical applications which produce a useful, concrete and tangible result. Specifically, the claimed member recognizer configured to recognize a member function from a family of related functions, the family start caller configured to make a family-start function call, and the member finish caller to make a member-finish function call, *taken as a whole*, produce a useful, concrete and tangible result, i.e. *a compiler used by a computer architecture to compile a family of related functions*. In addition, the claimed compiler is tangibly embodied and executed by a piece of hardware, i.e. the compiler is used by a computer architecture which is a tangible piece of hardware.

Furthermore, the functions of the claimed components of the compiler, i.e. the member recognizer, the family start caller, and the member finish caller, *taken as a whole* also accomplishes a practical application which produces a useful, concrete and tangible result as required under *State Street*. Specifically, the claimed components function “to compile a family of related functions” as *used by a computer architecture*.

For at least the reasons set forth above, the Applicants submit that claims 1-12 are directed to statutory subject matter, and thus comply with the requirements of 35 U.S.C. § 101. Applicants therefore respectfully request withdrawal of the rejection.

*Claim Rejection under 35 U.S.C. 102*

The test for determining if a reference anticipates a claim, for purposes of a rejection under 35 U.S.C. § 102, is whether the reference discloses all the elements of the claimed combination, or the mechanical equivalents thereof functioning in substantially the same way to produce substantially the same results. As noted by the Court of Appeals for the Federal Circuit in *Lindemann Maschinenfabrick GmbH v. American Hoist and Derrick Co.*, 221 USPQ 481, 485 (Fed. Cir. 1984), in evaluating the sufficiency of an anticipation rejection under 35 U.S.C. § 102, the Court stated:

Anticipation requires the presence in a single prior art reference disclosure of each and every element of the claimed invention, arranged as in the claim.

Therefore, if the cited reference does not disclose each and every element of the claimed invention, then the cited reference fails to anticipate the claimed invention and, thus, the claimed invention is distinguishable over the cited reference.

**PATENT**

Atty Docket No.: 10008025-1  
App. Ser. No.: 09/964,769

The Office Action sets forth a rejection of claims 1-3, 6-9, 12-15, 18-21 and 24 under 35 USC 102(e) as being allegedly anticipated by U.S. Patent No. 6,567,831 to Loginov ("Loginov"). This rejection is respectfully traversed.

Loginov teaches a method for compiling function evaluations performed by a Very Long Instruction Word (VLIW) processor using parallel computing algorithms (Abstract; Column 2, lines 38-58). According to Loginov, a VLIW processor uses parallel algorithms to minimize computation times (Col. 1, lines 24-38). Loginov teaches that, by moving scheduling tasks to a compiler, in which the compiler performs most instruction scheduling at compile time rather than execution units performing scheduling tasks at runtime, a VLIW processor avoids complex circuitry associated with on-chip instruction scheduling logic (Col. 1, lines 34-43). Based on the instruction scheduling performed by the compiler, the processing performed by the VLIW processor architectures is improved.

Loginov also teaches that the compiler performs function evaluations. In Loginov, a function, such as  $\sqrt{x}$ , is evaluated by representing the function as a series expansion, such as shown in column 3, lines 15-20. Values for one part of the series expansion of equation 1 are computed and stored in a table. The remaining part of the series expansion is a polynomial, as shown in lines 25-30. The polynomial can be quickly computed using parallel computation schemes, which greatly reduces processing time. (Col. 3, lines 45-55).

According to an embodiment of Applicants' invention, a compiler is provided to optimize the compiling of a family of related functions. A member recognizer is configured to recognize a member function from the family of related functions and a family start caller is configured to make a family-start function call for the family of functions related to the member function. For example, the following statements may appear in a computer program:

$X = \sin(\theta);$

$Y = \cos(\theta);$

The statement  $X = \sin(\theta)$  would be recognized as being a member of a known family of related functions, namely the trigonometric family of functions. This step of recognizing the statement  $X = \sin(\theta)$  as a member function of the trigonometric family of functions is performed by a member recognizer configured to recognize a member function from the family of related functions. A family start caller, configured to make a family-start function call for the family of functions related to the member function, then makes a family-start function call for the trigonometric family of functions related to the recognized member function, specifically the member function recognized by the statement  $X = \sin(\theta)$ . A member finish caller then makes a member-finish function call for the recognized member function, i.e. the statement  $X = \sin(\theta)$ . The compiler thus recognizes the member function and performs the appropriate family-start and member-finish function calls to give the following result for the  $X = \sin(\theta)$  member function:

$R1 = \text{call\_trigstart}(\theta);$

$R2 = \text{call\_sinfinish}(R1);$

The call instruction  $R1 = \text{call\_trigstart}(\theta)$  is the appropriate family-start function call for the trigonometric family of functions, and the call instruction  $R2 = \text{call\_sinfinish}(R1)$  is the appropriate member-finish function call for the  $X = \sin(\theta)$  member function.

The program statement  $Y = \cos(\theta)$  would also be recognized, by a member recognizer configured to recognize a member function from the family of related functions, as being a member of the same trigonometric family of functions. A family start caller, configured to make a family-start function call for the family of functions related to the member function,

then makes a family-start function call for the trigonometric family of functions related to the recognized member function  $Y = \cos(\theta)$ . A member finish caller then makes a member-finish function call for the recognized member function  $Y = \cos(\theta)$ . The compiler thus recognizes the member function and performs the appropriate family-start and member-finish function calls to give the following result for the  $Y = \cos(\theta)$  member function:

```
R3 = call_trigstart(theta);
```

```
R4 = call_cosfinish(R3);
```

The call instruction  $R3 = \text{call\_trigstart}(\theta)$  is the appropriate family-start function call for the trigonometric family of functions, and the call instruction  $R4 = \text{call\_cosfinish}(R3)$  is the appropriate member-finish function call for the  $Y = \cos(\theta)$  member function.

Prior to optimizing the function calls by the compiler, the program statements are:

```
R1 = call_trigstart(theta);
```

```
R2 = call_sinfinish(R1);
```

```
R3 = call_trigstart(theta);
```

```
R4 = call_cosfinish(R3);
```

Thus, the  $\text{call\_trigstart}(\theta)$  call is performed twice. According to an embodiment, all instructions, including the family-start calls  $\text{call\_trigstart}(\theta)$ , are subject to optimization techniques, including for example elimination of code that is redundant, by common subexpression elimination, code motion, and dead-code elimination techniques. An elimination routine would recognize that  $R1$  and  $R3$  are identical, and would eliminate the repetitive code, transforming the above code to look like:

```
R1 = call_trigstart(theta);
```

```
R2 = call_sinfinish(R1);
```

R4 = call\_cosfinish(R3);

Thus, the compiler operates to reduce the total number of instructions to be performed. For example, if R1 and R3 each comprised the *identical* 48 instructions, then eliminating R3 by implementing an optimizing technique translates to elimination of 48 instructions to be completed by the computer architecture. This elimination of redundant or repetitive instructions by the compiler, as illustrated by the example discussed above, enhances the processing speed by the computer architecture using the compiler, and also reduces the time and resources necessary for generating an executable file. Furthermore, any combination of a member recognizer, a family start caller, and a member finish caller may be incorporated into a front end of the compiler. Although the above example discusses the trigonometric family of functions, the compiler may compile any family of related functions.

Claim 1 recites a compiler to compile a family of related functions, comprising “a member recognizer configured to recognize a member function from said family of related functions,” “a family start caller configured to make a family-start function call for said family of related functions” and “a member finish caller to make a member-finish function call for said member function.” Claim 13 recites a method to compile a family of related functions comprising “recognizing a member function from said family of related functions”, “making a family-start call for said family of related functions” and “making a member-finish call for said member function.”

Loginov fails to teach “a member recognizer configured to recognize a member function from said family of related functions,” as recited in claim 1. Specifically, the compiler taught by Loginov fails to recognize a member function from a family of related functions. Instead, Loginov individually evaluates a function, such as  $\text{sqrt}(x)$ , by expanding



the function into a series expansion and performing parallel processing on a portion of the series expansion to improve processing time.

The rejection cites column 2, lines 52-55 and paraphrases this passage stating “overall improvement in processing speed in the evaluation of certain (families of) functions is achieved by (recognizing a member function from a family of related functions, then) representing each function as a series expansion.” It appears the rejection is alleging that a function disclosed in Loginov, such as  $\sqrt{x}$ , is the claimed family of related functions, and the series expansion disclosed by Loginov includes recognizing a member function from a family of related functions.

The rejection appears to be unsupported by the disclosure of Loginov. Specifically, Loginov discloses and describes  $\sqrt{x}$  and other functions as a single function and not a family of related functions. Furthermore, the series expansion disclosed by Loginov does not recognize a member function from a family of related functions. Instead, the series function is simply another representation of the function, such as shown in equation 1 of Loginov. Thus, Loginov fails to teach a member recognizer configured to recognize a member function from a family of related functions. Loginov also fails to teach a family of related functions or recognizing a specific function from the family. Loginov does not disclose that  $\sqrt{x}$ ,  $\sqrt[3]{x}$ , and  $\ln(x)$  are part of a family of related functions or only applying the function evaluation for these functions.

Loginov also fails to teach “a family start caller configured to make a family-start function call for said family of related functions” and “a member finish caller to make a member-finish function call for said member function.” Loginov teaches that parallel algorithms are used to compute “expansion series,” wherein each function is represented “as

a series expansion around one or more function argument values.” (Column 2, lines 50-58).

In the passage corresponding to column 2, line 66 through column 3, line 1, as cited by the Examiner, Loginov teaches “the first step of the method...is to divide the range of argument values for the approximation into n intervals.” However, Loginov fails to teach that either an expansion series or a range of argument values is a “family of related functions,” as recited in claim 1. Loginov also fails to teach that dividing a range of argument values for an approximation into n intervals is identical to making a family-start function call.

Furthermore, Loginov fails to teach that either the parallel algorithms, the compiler itself, the table approximation, the VLIW processor, or any other component of the system taught by Loginov includes “a member finish caller to make a member-finish function call for said member function,” as instantly claimed. Nowhere does Loginov teach that a compiler makes a member-finish function call. The compiler taught by Loginov performs function evaluations, but does not make a member-finish function call or any call for that matter. For at least the same reasons, Loginov also fails to teach “making a member-finish call for said member function,” as recited in claim 13.

Because claims 2-3, 6-9 and 12 incorporate all the limitations of claim 1, and because claims 14-15, 18-21 and 24 incorporate all the limitations of claim 13, Loginov fails to teach the invention claimed in claims 1-3, 6-9, 12-15, 18-21 and 24 for at least the reasons given above. Therefore, Loginov does not anticipate the subject matter of claims 1-3, 6-9, 12-15, 18-21 and 24. Claims 1-3, 6-9, 12-15, 18-21 and 24 are thus allowable over Loginov, and withdrawal of the rejection is respectfully requested.

*Claim Rejection Under 35 U.S.C. §103*

The test for determining if a claim is rendered obvious by one or more references for purposes of a rejection under 35 U.S.C. § 103 is set forth in MPEP § 706.02(j):

To establish a *prima facie* case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. The teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art and not based on applicant's disclosure. *In re Vaeck*, 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991).

Therefore, if the above-identified criteria are not met, then the cited reference(s) fails to render obvious the claimed invention and, thus, the claimed invention is distinguishable over the cited reference(s).

The Office Action sets forth a rejection of claims 4, 5, 10, 11, 16, 17, 22 and 23 under 35 U.S.C. § 103(a) as allegedly being unpatentable over Loginov in view of Aho. This rejection is respectfully traversed.

Neither Aho nor Loginov teach or suggest the features of independent claims 1 and 13, and thus dependent claims 4, 5, 10, 11, 16, 17, 22 and 23 are believed to be allowable. Furthermore, it would not have been obvious to combine Aho with Loginov. With regard to claim 4, Aho was cited in the rejection to teach that an optimizer is configured to in-line expand at least one of said family-start or member-finish calls. However, Aho discloses in-line expansion may be used for reducing the running time of a program (page 428, Aho), in the context of in-line expansion in call-by-name procedures (pages 428) in a run-time environment. However, Loginov discloses function evaluation performed in a compile-time environment and not a run-time environment. Thus, it would not have been obvious if not

possible to perform the in-line expansion described in Aho in the compile-time environment for the VLIW processor of Loginov.

With regard to claim 5, the Examiner paraphrases page 592, lines 10-12 of Aho, and states that Aho teaches subexpression elimination, code motion, and dead-code elimination as common examples of function preserving transformations. However, Aho fails to teach or suggest that subexpression elimination, code motion, and dead-code elimination may be used as function preserving transformations specifically by a compiler configured to optimize at least one of said family-start or member-finish calls, as instantly claimed.

With regard to claim 10, the Examiner paraphrases page 463, lines 1-3 of Aho, and states that Aho teaches making function calls in an intermediate language. However, Aho fails to teach a compiler used by a computer architecture to compile a family of related functions, comprising “a member recognizer configured to recognize a member function from said family of related functions,” “a family start caller configured to make a family-start function call for said family of related functions” and “a member finish caller to make a member-finish function call for said member function,” wherein one or both of said family start caller and said member finish caller are configured to make said family-start and member-finish function calls, respectively, in an intermediate language. Furthermore, with regard to claim 11, Aho also fails to teach or suggest that said intermediate language is non-architecture specific and non-operating system specific. Although the Examiner paraphrases page 463, lines 1-3 of Aho, nowhere does Aho specifically teach that the front end of a compiler translates a source program into *an non-architecture specific and non-operating system specific* intermediate language representation from which the back end generates

target code. Since Aho also fails to remedy the deficiencies of Loginov, the Applicants respectfully request withdrawal of the rejection.

With regard to claims 16, 17, 22 and 23, the Examiner only alleges that “the Loginov/Aho combination also discloses such claimed limitations as addressed in claims 4, 5, 10 and 11, respectively.” However, the Examiner provides no specific teaching or suggestion by Loginov or Aho, whether alone or in combination, to support the rejection. For at least the reasons discussed above, with regard to claims 4, 5, 10 and 11, respectively, the Applicants respectfully submit that neither Loginov or Aho, whether alone or in combination, teach or suggest the subject matter of claims 16, 17, 22 and 23.

Furthermore, Aho was cited merely to provide definitions of certain terms, e.g. call-by-name procedures (pages 428), function preserving transformations (page 592) and intermediate code generation (page 463). However, Aho describes these terms in the context of a run-time environment, and not in the context of a compiler. Because the instant claims pertain to a compiler used by a computer architecture to compile a family of related functions, and not to a run-time environment, the Applicants submit that Aho fails to provide any teaching or suggestion to render the claims obvious and it would not have been obvious to combine Aho with Loginov.

#### Newly Added Claims

Claims 25-28 have been added. Claims 25 and 26 each depend from allowable claim 1 and are also allowable at least by virtue of their dependency. Claims 27 and 28 each depend from allowable claim 13 and are allowable at least by virtue of their dependency. Claims 25 and 27 each recite “wherein at least one calculation is almost identical for each

**PATENT**

Atty Docket No.: 10008025-1  
App. Ser. No.: 09/964,769

member function of the family of related functions”, which is not taught or suggested by the prior art. Furthermore, the prior art fail to teach or suggest “wherein at least one calculation is identical for each member function of the family of related functions”, as recited in each of claims 26 and 28. Therefore, the Examiner is respectfully requested to allow claims 25-28.

**PATENT**

Atty Docket No.: 10008025-1  
App. Ser. No.: 09/964,769

Conclusion

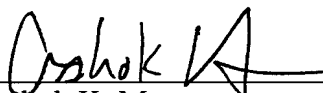
In light of the foregoing, withdrawal of the rejections of record and allowance of this application are earnestly solicited. Should the Examiner believe that a telephone conference with the undersigned would assist in resolving any issues pertaining to the allowability of the above-identified application, please contact the undersigned at the telephone number listed below. Please grant any required extensions of time and charge any fees due in connection with this request to deposit account no. 08-2025.

Respectfully submitted,

Peter Markstein

Dated: November 3, 2004

By

  
\_\_\_\_\_  
Ashok K. Mannava

Registration No. 45,301

MANNAVA & KANG, P.C.  
8221 Old Courthouse Road  
Suite 104  
Vienna, VA 22182  
(703) 652-3822  
(703) 880-5270 (facsimile)